IP Handling Update

IETF 102

Justin Uberti, Youenn Fablet

Recap of Key Goals

For arbitrary web pages (i.e., those without consent), prevent

1) **Detection of VPNs** (from exposing public IPs that aren't already visible)
2) **Fingerprinting/supercookie** using private IPs

while maintaining the ability to establish direct peer-to-peer connections.

Data Channel Usage

Why is allowing p2p in an arbitrary web page important?

Chrome data: **5x** as many web pages use data channels as audio/video; traffic growing **400%** y/y.

Problem 1: VPN Detection (Solved)

Key issue: detection of ISP public IP when using a split-tunnel VPN

Solved by forcing WebRTC to use the same network interface as for HTTP traffic; solution deployed successfully

Problem 2: Private IPs

Key issue: to allow direct connections, WebRTC impls expose the 'local' IP of the selected interface

Previous discussion considered this a minor issue, because:

- Allowing direct connections (i.e., no TURN) is important
- Private IPv4s usually have low entropy (e.g., 192.168.1.x)
- RFC4941 IPv6s are high entropy but short-lived (and perhaps public)

However:

- Not all private IPv4s are low-entropy
- Ongoing efforts to reduce fingerprinting surfaces
- Still a non-trivial number of web pages with 'suspicious' use of WebRTC

Safari Behavior

Safari took a more restrictive approach than other browsers - by default, **only IPs that are already visible to the web site are exposed.**

As a result, local IPs are not provided to the application; Safari-Safari direct connections not always possible

The team received feedback from a number of broken applications, including games and file transfer apps

mDNS Solution

Proposed solution from the Safari team:

- When gathering ICE candidates, replace private IPv4s with mDNS hostnames of the form <uuid>.local
- When receiving a .local ICE candidate, do mDNS resolution to obtain IP
- Direct connections, no private IPs exposed to apps!

```
candidate:1 1 udp 2113929471 B55ACF61-E9D1-4CD2-BA5C-22621A1F2F14.local 10000 typ host
```

instead of

~~`candidate:1 1 udp 2113929471 192.168.1.7 10000 typ host`~~

Analysis

Works well for previously broken use cases on unmanaged networks

Adds some potential latency (mDNS resolution) to connection setup

Does not always work for enterprise networks

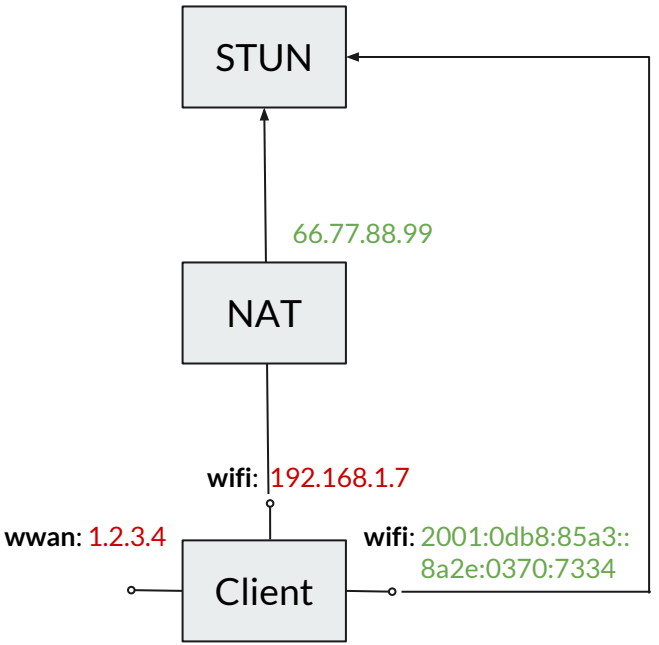
- When multicast is disabled
- On large intranets

Hybrid Solution

Proposed Hybrid Solution:

- Expose IPv4/IPv6 if public*
- Use mDNS to hide private addresses
- Need for mDNS goes away as IPv6 deployment proceeds

```
candidate:1 1 udp 2213929471 B55ACF61-E9D1-4CD2-BA5C-22621A1F2F14.local 10000 typ host
candidate:1 1 udp 2113929471 2001:0db8:85a3::8a2e:0370:7334 10000 typ host
candidate:1 1 udp 2003929471 66.77.88.99 12345 typ srflx raddr 0.0.0.0 rport 0
```



Consensus Call

Should we update the IP handling document to:

- Explicitly try to solve the private IP issue
- Use a mDNS-based technique to do so
- Make this technique the new default behavior

Private IPv6 Addresses

Some IPv6 addresses may not be public (e.g. NAT64 addresses)

- Detection requires deployment of IPv6-enabled STUN servers (uncommon)
- Even once deployed, some remaining impact on connectivity

Planning to run experiments to better understand potential impact and path forward

Context Linkage

- WebRTC allows connections between pages in different browsing contexts (e.g., between a normal page and a page in a private browsing tab)
- When this occurs via a host-host connection, the resultant low RTT can be used to infer that these pages are on the same host (with some false +/-)
- Already possible through various means, e.g. public IP + user agent string (OS, browser, version)
- No clear solution for IPv6

Should we document this issue?

Should we try to solve it?